THE RPR-FOM
A REFERENCE FEDERATION OBJECT MODEL TO PROMOTE SIMULATION INTEROPERABILITY

Graham C. Shanks
GEC Marconi S3I
Simulation and Training Division
Donibristle Industrial Park
Nr Dunfermline
Fife, Scotland KY11 5JX
graham.shanks@gecm.com

KEYWORDS
Reference FOM, Federation design, Interoperability

ABSTRACT

One of the goals of the High Level Architecture (HLA) is to promote interoperability, yet no constraints are put upon the federation designer during the Federation Object Model (FOM) development process. Unless federation designers cooperate to produce compatible FOMs then federates from different federations will not be able to interoperate with each other without extensive modifications, and the interoperability goal will not be achieved. It is unlikely, and perhaps even undesirable, to have a single FOM to cover all federations but each community served by HLA should be able to agree upon a common framework to cover their federations. Such a framework is called a Reference FOM, and federation designers would derive their FOM from it, extending and configuring it as required.

This paper describes the Real-time Platform Reference FOM (RPR-FOM), previously called the PnP-FOM (Plug 'n Play FOM). The RPR-FOM is intended to be a Reference FOM for real-time, platform level federations. Its development has been motivated by a desire to preserve and build upon the degree of interoperability that currently exists within the Distributed Interactive Simulation (DIS) community and to ease the transition of DIS simulations to HLA. It is built around the functionality contained within the IEEE 1278.1-1995 and DIS version 2.1 standards.

1. INTRODUCTION

The High Level Architecture (HLA) Management Plan[1] explicitly states that the purpose of the architecture is to facilitate interoperability among simulations and promote reuse of simulations and their components. However, HLA only directly addresses interoperability at the most basic level, by providing the Run Time Infrastructure to standardize the interface to the data interchange communications layer. Support for interoperability at higher levels is more indirect. The federation system designer is provided with a very flexible, but therefore potentially non-interoperable, design and documentation method via the Federation and Simulation Object Models (the FOM and SOM) with which to express the distributed system design. The object models provide some support for the designer to lay down some interoperability standards, by defining accuracy requirements for instance, but the choice of object decomposition and data representations is not constrained by the HLA specifications.

If federation system designers do not reach some consensus on object decompositions and data representations then the interoperability and reuse goals of the architecture will not be realized. If designers do not take into account these goals, and build their object models with interoperability in mind, then the federation will not be able to interoperate with other federations, and the potential for reuse of its constituent federates, without extensive modifications, will be greatly diminished.

One of the best examples of a community that has managed to achieve a workable level of interoperability is the Distributed Interactive Simulation (DIS) community. However, DIS only supports part of the modeling and simulation spectrum and has an inflexible set of data representations. Nevertheless there are many successful DIS simulation systems in existence, and many examples of simulation reuse within the community.

One of the reasons why the DIS community has managed to achieve these interoperability levels is that it has reached a consensus on a common object and interaction decomposition and a common set of data representations. In many cases this consensus is a compromise, where a data representation is selected primarily because it provides an interchange format between two, or more, common formats. For example, many simulations perform positional updates in a local coordinate system, converting to and from the DIS geocentric coordinate system when manipulating DIS Protocol Data Units (PDUs).

The strong desire of the DIS community to retain, and build upon, this level of interoperability within HLA has led to the development of the concept of Reference FOMs, and the creation of the Real-time Platform Reference FOM (RPR-FOM) to ease the transition for DIS compatible simulations to HLA.

2. REFERENCE FOMs

The idea behind a Reference FOM is that it can be used as a basis for both designing a federation and designing a federate. Using a Reference FOM as a basis for a federation means that there is likely to be a larger range of existing federates available for reuse. It also allows general purpose federates to be designed against the Reference FOM with a high probability that they can be used repeatedly. Of course, in order to be useful, a Reference FOM must address an area of interest to a user community.

Just like there can never be a single FOM that caters for all modeling and simulation applications, it is expected that there will be multiple Reference FOMs, each catering for one or more of the user domains. Some of the Reference FOMs may even overlap, requiring the members of each community to cooperate in reaching a consensus about the common areas, thereby facilitating simulator reuse between the two domains.

The main difference between a Reference FOM and a FOM is that a FOM is created with a specific software and hardware implementation in mind, i.e. the federation, whereas a Reference FOM is for a notional set of federations. To borrow an analogy from object orientated design, the Reference FOM is equivalent to a class definition, whilst a FOM derived from the Reference FOM is equivalent to a specific instance of that class. In order to use a Reference FOM the federation designer must instantiate it to produce the federation's FOM.

2.1 Instantiating a Reference FOM

Reference FOMs should follow all the requirements of the HLA Object Model Template[2], so that it is a valid FOM in its own right. Thus federation designers may choose to create a FOM for their federation by instantiating a Reference FOM without modification. However it is likely that many users of a Reference FOM will wish to use a modified instance of the Reference FOM to suit their particular needs. Federation designers, and their users, have to realize that by modifying the Reference FOM for their use they are weakening, and perhaps even destroying, the potential for reuse and interoperability with other simulations.

The main ways that a Reference FOM may be modified during instantiation are as follows:

- Subsetting
- Extension
- Overriding of default values
- Incorporation of FOM-lets

These methods are described in the following sections.

2.2 Reference FOM Subsetting

A Reference FOM may well contain objects, interactions and attributes that are not required by the federation. The Reference FOM should be structured so that it is easily decomposible so that the federation designer can readily construct the subset of the Reference FOM which is applicable to that federation. For instance the RPR-FOM contains Emitter System and Emitter Beam objects. If the federation being designed does not include any radar or ESM equipment then these objects are not required. The designer would instantiate the Reference FOM for such a federations by taking a copy of it and deleting these objects, and their associated attributes, from the copy.

FOM tools should provide support for subsetting. For instance, if an object is deleted from the object class structure table, then all references to it in other tables should also be deleted, any attributes associated with the class should be deleted, any complex data types or enumerations wholly associated with the attributes should be deleted (retaining any that are used by attributes of other object classes) and any interactions which require the deleted object class as either initiating object or receiving object should be deleted. Ideally this should happen automatically.

2.3 Reference FOM Extension

Even a well designed Reference FOM is unlikely to include all object classes, interactions and attributes required by all federations that wish to use it. In this case the federation designer would extend the Reference FOM by adding to the existing tables.

If the extension is a good, general purpose solution to a common problem then the extensions should be added to the Reference FOM itself, thereby making it available for future users of the Reference FOM. Failure to do this is likely to have an adverse effect on interoperability since other, incompatible solutions to the same problem may be proposed and implemented, leading to two distinct and non-interoperable factions within the target user community.

A particular instance of the need to extend a Reference FOM is the requirement to provide data specifically for exercise control or data logging purposes. For instance, the RPR-FOM only defines those attributes which are required by other simulation objects within the execution (based upon experience gained from DIS). It excludes “internal” attributes that cannot be sensed by other objects or effect other objects. For example fuel and ammunition levels are excluded. However these attributes may be required for exercise control and assessment purposes or by simulations used for test and evaluation. In order for execution control federates to gain access to these attributes they must be contained within the federation’s FOM. The potentially huge number of such attributes that could be required for control or logging purposes preclude them from being included in the base Reference FOM.

Since subsetting is an inherently easier process than extension (because subsetting involves making a value judgment on whether a feature is required or not, whereas extension usually involves an element of design), a good Reference FOM should be as complete as possible.

2.4 Overriding of default values

A Reference FOM contains a number of numeric values, primarily the accuracy and resolution values in the attribute/parameter and complex data type tables, but possibly in other areas, such as part of the update conditions in the attribute/parameter table. In a Reference FOM these values are treated as defaults. When instantiating the Reference FOM the federation designer should ensure that these values are valid for their federation, and override any values that are invalid.

Federations that choose accuracy and resolution values that differ greatly from the default values contained in the Reference FOM are likely to reduce the potential for interoperability. Designers should take this into account when deciding whether the default values should be overridden.

2.5 Incorporation of FOM-lets

Not all possible modifications to a Reference FOM are covered by the above instantiation methods. The need for more complex modifications has given rise to the concept of FOM-lets. FOM-lets are roughly equivalent to macros, in that they provide a series of instructions for modifying the Reference FOM during instantiation. FOM-lets would be used for more complicated, but still well defined, instantiation modifications.

One common reason for creating a FOM-let would be to implement functionality that is incompatible with the base functionality contained in the Reference FOM. Another reason would be where there were multiple, but incompatible, solutions to a problem. A good example of this would be dynamic terrain. There are a number of different architectural solutions for dynamic terrain, each with different advantages and disadvantages. A federation will choose the solution that best meets its requirements. The data to be passed between the various dynamic terrain elements depends mainly on the particular architecture chosen. There will be some commonality, for instance common data representations of terrain surfaces, but this may well be only a small proportion of the overall data definition. Each separate dynamic terrain architecture would give rise to a separate FOM-let, which contains the instructions for adding the dynamic terrain objects, interactions, attributes and update conditions specific to that architecture.

Other FOM-lets could extend the functionality of the base FOM for experimental reasons, often in a fairly narrow domain. For example, experimental smoothing algorithms, and the data to support them, may form a FOM-let. Most applications would not use it, but encapsulating the behaviour in a FOM-let ensures that the experience is not lost, and allows other federations to use it if they so desire.
Federations which incorporate mutually incompatible FOM-lets constrain their potential for interoperability and reuse, often quite severely.

3. THE RPR-FOM

The Real-time Platform Reference FOM (RPR-FOM) is specifically designed to maintain and build upon the interoperability achieved by DIS simulations within the HLA framework. It aims to ease the transition of DIS compatible simulations to HLA by providing a mapping from DIS functionality to HLA. Although the RPR-FOM arose out of the DIS++ transition efforts it should be able to be used as a basis for non-DIS federations and federates. Indeed in time, the RPR-FOM should be extended to cover functionality not contained in the current DIS standards.

Not all DIS functionality maps directly to elements of the HLA object models. Some DIS functionality, for instance the simulation management data query mechanism, is provided directly by the RTI[3]. In other cases the RTI provides only a partial solution to the DIS functionality, for instance the RTI can pause the execution (or freeze an exercise in DIS terminology) but cannot pause individual objects. The RPR-FOM excludes DIS functionality that doesn’t map directly to the HLA object model. It is intended that a guidance document be produced to provide the mapping from DIS to RTI functionality.

The title RPR-FOM should not be construed to mean that it is limited to either the real-time or platform domains. In exactly the same way as DIS can be used for non real-time applications and, with the introduction of Aggregate PDUs and IsPartOf PDUs within DIS version 2.1.4[5], for representing entities that are not individual platforms, then the RPR-FOM can support such applications. Instead, the terms reflect the “center of mass” of the user community that the RPR-FOM aims to support.

3.1 Examples from the RPR-FOM

Table 1 shows the RPR-FOM object class structure. It illustrates one of the early design decisions, that is the choice of a shallow class hierarchy over a deep class hierarchy.

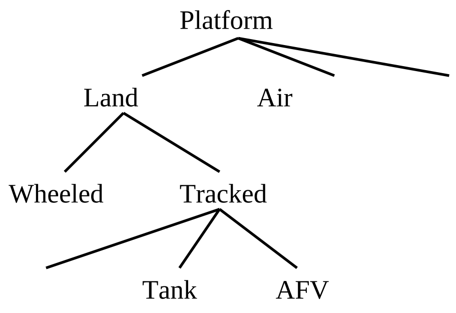

Figure 1 - Example of a deep hierarchy

Deep class hierarchies, such as the example in figure 1, allow a fine granularity for the association of attributes with object classes. However, they can be difficult, if not impossible, to construct without multiple inheritance or without duplication of attributes down multiple branches of the object hierarchy. They also have the disadvantage of forcing federates to subscribe to a large number of object classes if they require all attributes of all object classes (not an uncommon requirement, especially for applications with visual simulations).
Shallow hierarchies, such as that shown in Table 1, are easier to construct and allow a better level of abstraction to occur, but have the disadvantage that objects must own attributes that do not always make sense. Table 2, which is part of the RPR-FOM attribute/parameter table, illustrates this. All physical entities, including soldiers, have ramps and tents! However, by selecting suitable default values for such attributes it is possible to avoid any major problems. For instance, all non-moving entities, including buildings, have zero velocity, whilst tents and ramps will never be deployed by entities that do not have them.

3.2 RPR-FOM Development Program

A program for the development of the RPR-FOM has been proposed. Initial development efforts will concentrate on capturing the functionality contained within the DIS IEEE standard[4], before moving on to the version 2.1.4 functionality[5], the latter intended to track this working draft through to the updated IEEE specification. Finally extensions will be added to the RPR-FOM to take full advantage of the new features that HLA provides. The first two issues will map the DIS functionality straight into HLA, without "fixing" problems or errors within DIS.

Base Class	**1st Subclass**	**2nd Subclass**	**3rd Subclass**
BaseEntity (S)	PhysicalEntity (PS)	MilitaryEntity (S)	MilitaryPlatformEntity (PS)
			MunitionEntity (PS)
			Soldier (PS)
	AggregateEntity (PS)		
	EnvironmentEntity (PS)		
AttachedSystem (N)	EmitterSystem (PS)		
	RadioTransmitter (PS)		
	RadioReceiver (PS)		
	Designator (PS)		
EmitterBeam (PS)	TrackJamBeam (PS)		

Table 1 - RPR-FOM Object Class Structure Table

Class	Attribute	Datatype
BaseEntity	AccelerationVector	AcceleratonStruct
	AngularVelocityVector	AngVelocityStruct
	DRAlgorithm	DRAlgorithmEnum
	EntityType	EntityTypeStruct
	IsFrozen	RTI_Boolean
	Orientation	OrientationStruct
	Position	PositionStruct
	VelocityVector	VelocityStruct
PhysicalEntity	ArticulatedPartsArray	ArticulatedPartStruct
	ArticulatedPartsCount	RTI_UShort
	DamageState	DamageStateEnum
	EngineSmokePresent	RTI_Boolean
	FlamesPresent	RTI_Boolean
	HasFuelSupplyCap	RTI_Boolean
	HasRecoveryCap	RTI_Boolean
	HasRepairCap	RTI_Boolean
	HatchState	HatchStateEnum
	Immobilized	RTI_Boolean
	LightsState	LightStateEnum
	Marking	MarkingStruct
	PowerPlantOn	RTI_Boolean
	RampDeployed	RTI_Boolean
	SmokePlumePresent	RTI_Boolean
	TentDeployed	RTI_Boolean
	TrailState	TrailStateEnum

Table 2 - Extract from RPR-FOM Attribute/Parameter Table

The draft program for RPR-FOM development is as follows:

IEEE 1278.1 Functionality	Spring 97
DIS 2.1.4 Functionality	Fall 97
HLA Extensions	Spring 98

Since work started on the RPR-FOM in Summer 1996 a number of drafts have been produced. The latest draft is available electronically from the Simulation Interoperability Standards Organization web site at http://siso.sc.ist.ucf.edu. Updates will be posted to this site.

3.3 RPR-FOM and Data Standards

By encapsulating the common terminology and concepts of the user community within the object model, the RPR-FOM is more likely to provide a useful abstraction of the problem domain and be applicable to many areas of modeling and simulation within that community. The DMSO Data Standardization Project is providing technical support to the RPR-FOM development effort[6], specifically to identify common terminology and data representations for use within the RPR-FOM, including the use of the HLA Data Dictionary.

4. CONCLUSION

The concept of a Reference FOM, which provides an object model definition for a notional class of federations, is a powerful method of defining standard object, interaction and data decompositions that can be used as a basis for designing both federations and federates. Use of Reference FOM facilitates interoperability between federates and even federations that comply with it, although it does not guarantee this. A base of federates compliant with a Reference FOM will also facilitate simulation reuse.

The RPR-FOM provides current DIS simulations with an easy transition path to HLA, whilst retaining, and building upon, the interoperability benefits gained under the DIS paradigm. It is planned that the RPR-FOM will contain all the current DIS functionality and, indeed, will ultimately be expanded to cover functionality not currently contained within DIS.

5. ACKNOWLEDGMENT

The decision to develop the RPR-FOM was taken by the DIS++ task group, set up by the Protocols Working Group of the DIS workshop, under the leadership of Richard Schaffer. This group, of which the author is a member, has been responsible for many of the ideas contained within this paper (although the author bears the responsibility for any errors). It is proposed that this group continue the work of developing the RPR-FOM within the Simulation Interoperability Standards Organization (SISO).

6. REFERENCES

[1] HLA Management Plan, Defense Modeling and Simulation Office , Version 1.7, 1 April 1996.

[2] High Level Architecture, Object Model Template, Defense Modeling and Simulation Office , Version 1.0, 15 August 1996

[3] High Level Architecture, Interface Specification, Defense Modeling and Simulation Office , Version 1.0, 15 August 1996

[4] IEEE Standard for Distributed Interactive Simulation - Applications Protocols, IEEE Std 1278.1-1995

[5] Standard for Distributed Interactive Simulation - Applications Protocols, Version 2.1.4 (Working Draft), Institute for Simulation and Training, IST-CR-96-12

[6] HLA FOM/SOM Content Standards, Spring 97 Simulation Interoperability Workshop, 97S-SIW-180

7. ABOUT THE AUTHOR

GRAHAM SHANKS is the Chief Software Engineer for the Simulation and Training Division (North) of GEC Marconi S3I (GMS&T). He received a BSc (Hons) in Pure Mathematics from the University of Sheffield in 1979. He has over sixteen years experience in the real-time simulation of military equipment. He currently leads GMS&T's DIS and HLA technology research and development programmes and is an active participant in the DIS standards development process.